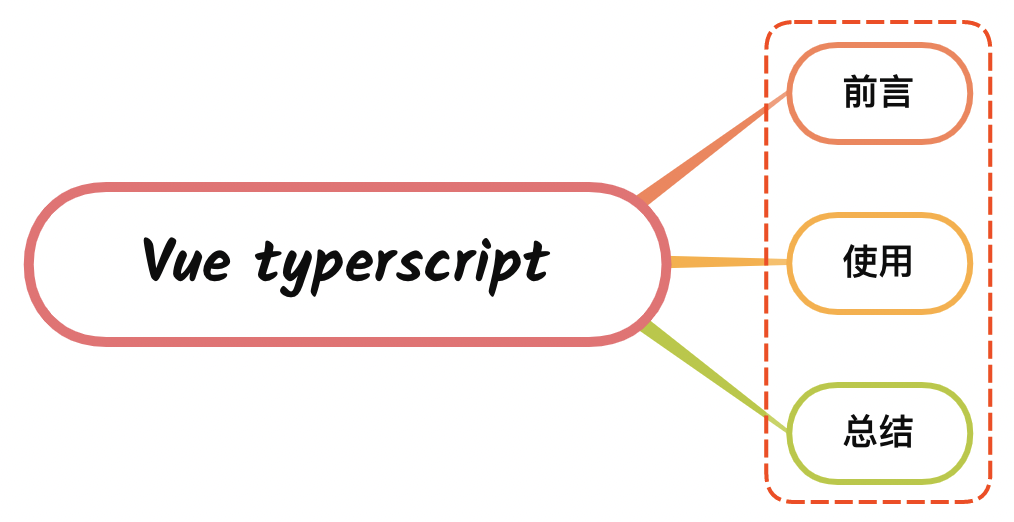
# 面试官：说说如何在Vue项目中应用TypeScript？



## 一、前言

与link类似

在VUE项目中应用typescript，我们需要引入一个库vue-property-decorator，

其是基于vue-class-component库而来，这个库vue官方推出的一个支持使用class方式来开发vue单文件组件的库

主要的功能如下：

* methods 可以直接声明为类的成员方法
* 计算属性可以被声明为类的属性访问器
* 初始化的 data 可以被声明为类属性
* data、render 以及所有的 Vue 生命周期钩子可以直接作为类的成员方法
* 所有其他属性，需要放在装饰器中

## 二、使用

vue-property-decorator 主要提供了多个装饰器和一个函数:

* @Prop
* @PropSync
* @Model
* @Watch
* @Provide
* @Inject
* @ProvideReactive
* @InjectReactive
* @Emit
* @Ref
* @Component (由 vue-class-component 提供)
* Mixins (由 vue-class-component 提供)

### @Component

Component装饰器它注明了此类为一个Vue组件，因此即使没有设置选项也不能省略

如果需要定义比如 name、components、filters、directives以及自定义属性，就可以在Component装饰器中定义，如下：

import {Component,Vue} from 'vue-property-decorator';  
import {componentA,componentB} from '@/components';  
  
 @Component({  
 components:{  
 componentA,  
 componentB,  
 },  
 directives: {  
 focus: {  
 // 指令的定义  
 inserted: function (el) {  
 el.focus()  
 }  
 }  
 }  
})  
export default class YourCompoent extends Vue{  
  
}

### computed、data、methods

这里取消了组件的data和methods属性，以往data返回对象中的属性、methods中的方法需要直接定义在Class中，当做类的属性和方法

@Component  
export default class HelloDecorator extends Vue {  
 count: number = 123 // 类属性相当于以前的 data  
  
 add(): number { // 类方法就是以前的方法  
 this.count + 1  
 }  
  
 // 获取计算属性  
 get total(): number {  
 return this.count + 1  
 }  
  
 // 设置计算属性  
 set total(param:number): void {  
 this.count = param  
 }  
}

### @props

组件接收属性的装饰器，如下使用：

import {Component,Vue,Prop} from vue-property-decorator;  
  
@Component  
export default class YourComponent extends Vue {  
 @Prop(String)  
 propA:string;  
  
 @Prop([String,Number])  
 propB:string|number;  
  
 @Prop({  
 type: String, // type: [String , Number]  
 default: 'default value', // 一般为String或Number  
 //如果是对象或数组的话。默认值从一个工厂函数中返回  
 // defatult: () => {  
 // return ['a','b']  
 // }  
 required: true,  
 validator: (value) => {  
 return [  
 'InProcess',  
 'Settled'  
 ].indexOf(value) !== -1  
 }  
 })  
 propC:string;  
}

### @watch

实际就是Vue中的监听器，如下：

import { Vue, Component, Watch } from 'vue-property-decorator'  
  
@Component  
export default class YourComponent extends Vue {  
 @Watch('child')  
 onChildChanged(val: string, oldVal: string) {}  
  
 @Watch('person', { immediate: true, deep: true })  
 onPersonChanged1(val: Person, oldVal: Person) {}  
  
 @Watch('person')  
 onPersonChanged2(val: Person, oldVal: Person) {}  
}

### @emit

vue-property-decorator 提供的 @Emit 装饰器就是代替Vue中的事件的触发$emit，如下：

import {Vue, Component, Emit} from 'vue-property-decorator';  
 @Component({})  
 export default class Some extends Vue{  
 mounted(){  
 this.$on('emit-todo', function(n) {  
 console.log(n)  
 })  
 this.emitTodo('world');  
 }  
 @Emit()  
 emitTodo(n: string){  
 console.log('hello');  
 }  
 }

## 三 、总结

可以看到上述typescript版本的vue class的语法与平时javascript版本使用起来还是有很大的不同，多处用到class与装饰器，但实际上本质是一致的，只有不断编写才会得心应手